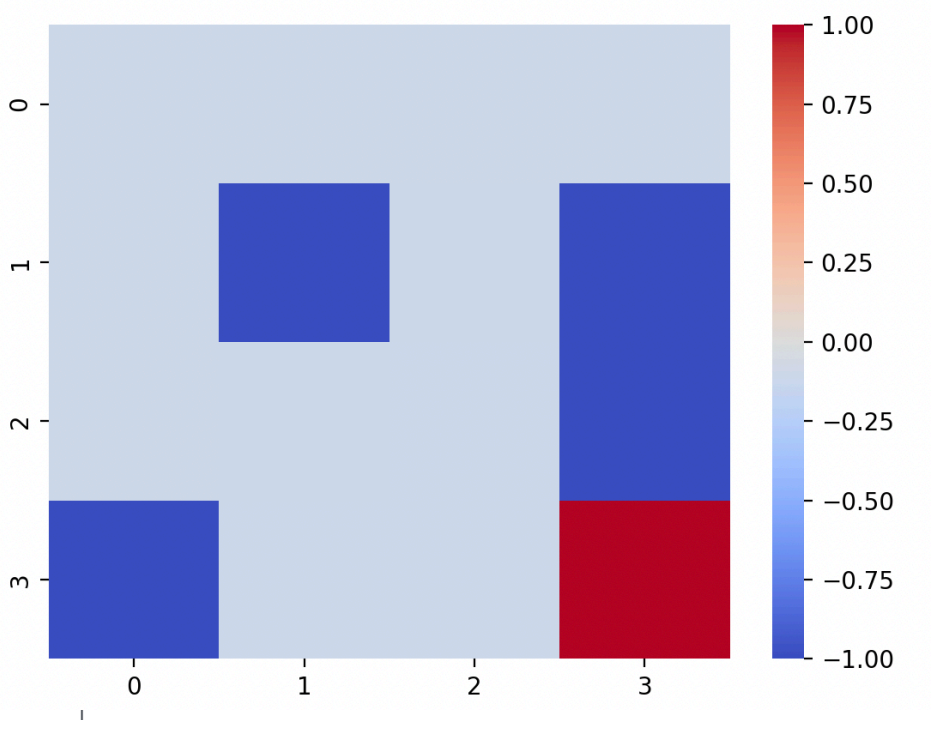
FAG ØVING

Utilities

Implementation of value iteration algorithm

```python
def find_utlities_using_value_iteration():
    """
    This function finds the utilities using the value iteration algorithm.
    :return: a numpy array of size 16.
    """
    utilities = np.zeros(N_STATES)
    delta = float('inf')
    while delta > EPSILON + 2:
        delta = 0
        for state in range(N_STATES):
            if not valid_state(state):
                continue
            max_utility = -np.inf
            for action in range(N_ACTIONS):
                utility = 0
                for next_state in get_next_states(state, action):
                    trans_prob = get_trans_prob(state, action, next_state)
                    utility += trans_prob * utilities[next_state]
                if utility > max_utility:
                    max_utility = utility
            delta = max(delta, abs(max_utility - utilities[state]))
            utilities[state] = get_reward(state) + GAMMA * max_utility
    return utilities
```

Displayed with heatmap function



Gready policy

Implementation of algorithm

```python
def find_greedy_policy(utilities: np.ndarray):
    """
    This function finds the greedy policy given the utilities.
    :param utilities: a numpy array of size 16.
    :return: a numpy array of size 16.
    """
    policy = np.zeros(N_STATES)
    for state in range(N_STATES):
        if not valid_state(state):
            continue
        max_utility = -np.inf
        for action in range(N_ACTIONS):
            utility = 0
            for next_state in get_next_states(state, action):
                trans_prob = get_trans_prob(state, action, next_state)
                utility += trans_prob * utilities[next_state]
            if utility > max_utility:
                max_utility = utility
                policy[state] = action
    return policy
```

Visualization